Computational Science Technical Note **CSTN-114**

Optimal Data Structures for Spatially Localised Agent-Based Automata and Hybrid Systems

C. J. Scogings and K. A. Hawick

2012

Agent-based systems and cellular automata are two closely related model formulations that are heavily used in studying complex systems. They are both formulated as microscopically simple rule-based models that are applied to individual cells or agents in a collection, where the spatially localised neighbourhood of other cells or agents is used as input to update each one. We have experimented with a range of models including classic cellular automata, through more sophisticated multi-state automata, flocking models, and stochastic-agent models and animat agent-based predator-prey models. We discuss algorithmic commonalities and code implementation patterns that have emerged as common properties of these models and describe how we have experimented with optimal data structures to support spatially localised models of this class. We show how concurrency and model correctness issues are affected by different data structures in addition to their effect on model update computational efficiency.

Keywords: agent-based models; cellular automata; hybrid model; spatial locality; rule-set; efficient update; neighbourhood

BiBTeX reference:

```
@INPROCEEDINGS{CSTN-114,
        author = {C. J. Scogings and K. A. Hawick},
        title = {Optimal Data Structures for Spatially Localised Agent-Based Automata
                and Hybrid Systems},
        booktitle = {Proc. Int. Conf. on Artificial Intelligence and Soft Computing},
        year = {2012},
        pages = {221-227},
        address = {Napoli, Italy},
        month = {25-27 June},
        publisher = {IASTED},
        institution = {Computer Science, Massey University},
        keywords = {agent-based models; cellular automata; hybrid model; spatial locality;
                rule-set; efficient update; neighbourhood},
        owner = {kahawick},
        timestamp = {2010.09.09}
}
```

Technical Report CSTN-114

Optimal Data Structures for Spatially Localised Agent-Based Automata and Hybrid Systems

C.J. Scogings and K.A. Hawick
Computer Science, Institute for Information and Mathematical Sciences,
Massey University, North Shore 102-904, Auckland, New Zealand
email: { c.scogings, k.a.hawick }@massey.ac.nz
Tel: +64 9 414 0800 Fax: +64 9 441 8181

March 2011

ABSTRACT
Agent-based systems and cellular automata are two closely related model formulations that are heavily used in studying complex systems. They are both formulated as microscopically simple rule-based models that are applied to individual cells or agents in a collection, where the spatially localised neighbourhood of other cells or agents is used as input to update each one. We have experimented with a range of models including classic cellular automata, through more sophisticated multi-state automata, flocking models, and stochastic-agent models and animat agent-based predator-prey models. We discuss algorithmic commonalities and code implementation patterns that have emerged as common properties of these models and describe how we have experimented with optimal data structures to support spatially localised models of this class. We show how concurrency and model correctness issues are affected by different data structures in addition to their effect on model update computational efficiency.

KEY WORDS
agent-based models; cellular automata; hybrid model; spatial locality; rule-set; efficient update; neighbourhood.

1 Introduction

Agent-based models have been used to make significant contributions in a variety of different disciplines including: Artificial Life [1, 18, 36]; ecosystems [29]; predator-prey systems [10,31]; trading and economics [8,22,24]; traffic and transport [6,25]; and military simulations [3,4,35].

Such spatial agent models are often known as "animat" models [38] and have several features and properties in common with classic cellular automata including:

- the grid (or map) – visually the action takes place in a particular area of space (often two-dimensional but three-dimensional models also exist, such as Boids).
- a sequence of "time steps" simulating the passage of time
- the application of rules to individual agents or cell locations
- the "local view" – agents react to neighbours within a reasonably small bounded area
- the emergence of apparently unpredictable macro-patterns and behaviours such as flocking [19,27] or spirals [14]
- other complex and adaptive (spatial) properties [5,32].

Craig Reynolds [26, 28] refers to agent-based models as "individual-based models" in which a number of individual agents are defined by rules and parameters. He distinguishes between cellular automata and individual-based models as follows:

- cellular automata – homogeneous and dense with an emphasis on cells in a grid. If neighbours are required then their cell locations are referred to explicitly

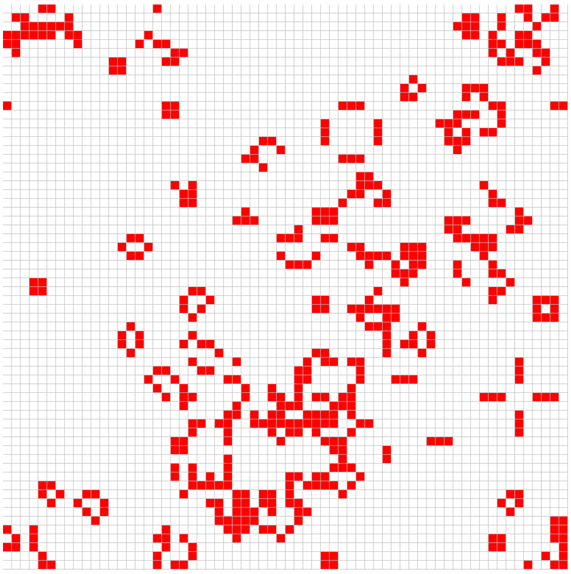

Figure 1: Conway's Game of Life is an example of a classic cellular automaton. During each time-step, every cell in the grid is checked and the rules are followed. Thus the rules are applied to the **grid location** and strictly autonomous agents do not exist.

- individual-based models – a number of individuals defined by rules and parameters. Individuals usually need to be aware of several "local" neighbours

Reynolds goes on to state that a major difference between these types of models is whether the simulation's inner loop proceeds cell by cell, or individual by individual.

In fact there are good reasons why an advanced agent-based model should combine the important elements of both cellular automata and individual-based models. This article introduces an agent-based predator-prey model [31] that demonstrates the efficiency of this combined approach.

Our article is structured as follows: Brief overviews of Cellular Automata (CA) and Individual-based Models are provided in sections 2 and 3 respectively. Our predator-prey model is introduced in section 4. The different approaches to the use of the grid and the search for neighbours are covered in section 5 and some conclusions are presented in section 7.

2 Cellular Automata

A cellular automaton (*pl:* automata) consists of a regular grid of cells where each cell is in one of a number of states. A new generation is created by applying a fixed rule to each cell to determine the new state of the cell. Cellular automata were developed at the Los Alamos National Laboratory in the 1940s by John von Neumann after a suggestion by his colleague Stanisław Ulam. Considerable work on CAs has been reported in the literature [40] including statistical mechanics [39] and use of cyclic rules [16, 17].

One well-known example of a cellular automaton is the Game of Life devised by John Conway in 1970 [7]. The Game consists of an infinite two-dimensional grid of square cells and each cell has two possible states, *live* or *dead*. At each step in time, every cell interacts with its eight immediately adjacent neighbours by applying the following rules:

1. Any live cell with less than 2 live neighbours dies.
2. Any live cell with greater than 3 live neighbours dies.
3. Any live cell with 2 or 3 live neighbours remains unchanged.
4. Any dead cell with 3 neighbours comes to life.

The Game is initialised with a pattern of live and dead cells. The first generation is created by applying the rules simultaneously to every cell and then the rules are applied repeatedly to subsequent generations. The Game has attracted much interest because of the surprising ways in which patterns can evolve. Many initial patterns along with various extensions can be found on a large number of web sites. An example is shown in Figure 1.

In each time step, every cell of the grid is checked in turn. Neighbours are defined by cell location. For example the adjacent neighbours of cell [9,6] are cells [8,5], [8,6], [8,7], [9,5], [9,7], etc. The key point about these CA models is the lack of autonomy. Each cell is updated deterministically and obeys its rules slavishly and in synchrony.

It is possible to introduce some degree of stochasticity into automata-like models. A classic case is the Ising model [20] which has a synchronously applied microscopic rule but where a thermal heat-bath in the form of a source of pseudo randomly generated numbers is used to allow for some cells to increase their energy instead of monotonously decreasing their energy and slavishly obeying their update rule.

The Monte Carlo method is usually employed to evolve the Ising system in a sort of "pseudo-time". A typical quenching simulation or experiment is therefore to initialise an Ising spin variable (dumb agent) on each site. Spin sites are updated depending upon the energy consequences of the updates. In the Ferromagnetic variation of the model it is energetically favourable for spins to align. The Metropolis [23] algorithm for the Ising system is therefore:

1. pick a spin site
2. compute the change in the number of like-like bondsΔ with its neighbours if the spin value flips
3. if the change Δ is positive or zero accept the change and flip the spin
4. if the change Δ is negative accept it with Probability $P = e^{-2\Delta K}$

This procedure is carried out for each spin site, a full sweep constituting on average one "hit" per site. This conceptualisation of an Ising dumb spin agent updating itself on the basis of its neighbouring agents is shown in Figure 2

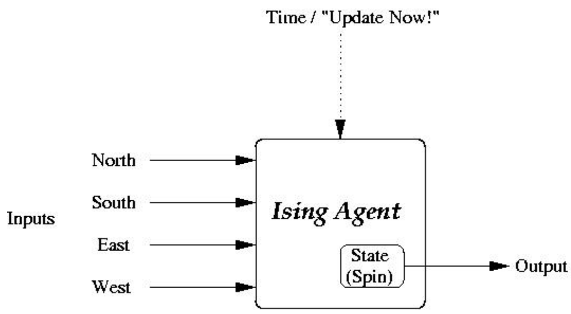

Figure 2: An "Ising Agent" operating on a 2-d Square lattice with 4 definite inputs from its neighbouring spins and an implied synchronizing input specifying when it should update its state.

Similar ideas are sometimes applied to opinion formation models such as Sznajd's model [33] which has many similarities to the Ising system, except that information is propagated outwards to neighbouring agents [15], instead of inwards in the Ising case [34]. We have described the agent-like role of Ising and related systems in more detail elsewhere [9]. In this present paper we expand the spatial comparison to that of other multi-agent like system such as an individual-based system such as Reynolds' "Boids."

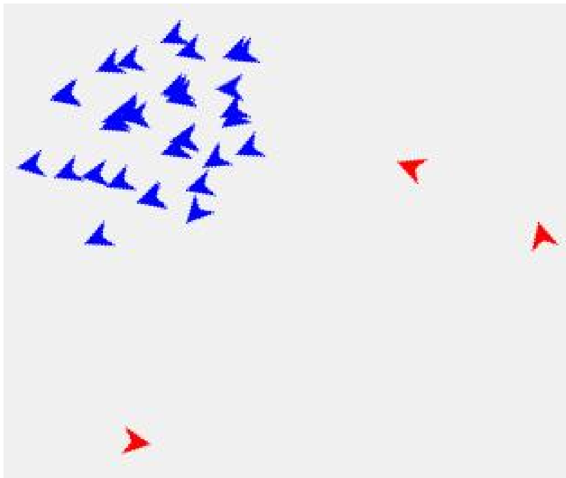

Figure 3: Craig Reynolds' Boids is an example of an individual-based spatially explicit model that uses continuous coordinates. During each time-step, each **agent** is updated autonomously.

3 Individual-Based Models

Most agent-based models will fit under the heading of Individual-Based Models. Usually there exist a large number of individual agents, each comprising a **current state** (eg: location, status, health, age) and a **set of rules**. Each time step, every agent is checked, the rules are applied and the state is updated.

For example, if a rule causes an agent to move, then the state is updated to reflect the new location. It is sometimes possible to have states being updated even if no rule is applied, for example: if a predator agent "eats" a prey agent the status of the prey agent is updated to "dead".

A well-known example is the three-dimensional model of coordinated animal motion called Boids [26]. The model simulates flocking behaviour by applying three simple steering behaviour rules to each agent. These rules are:

- separation – steer to avoid crowding local agents
- alignment – steer towards the average heading of local agents
- cohesion – steer towards the average position of local agents

More complex rules can be added such as avoiding obstacles and seeking goals. The basic model has been modified extensively with additions such as predator-prey situations and leadership. Observers

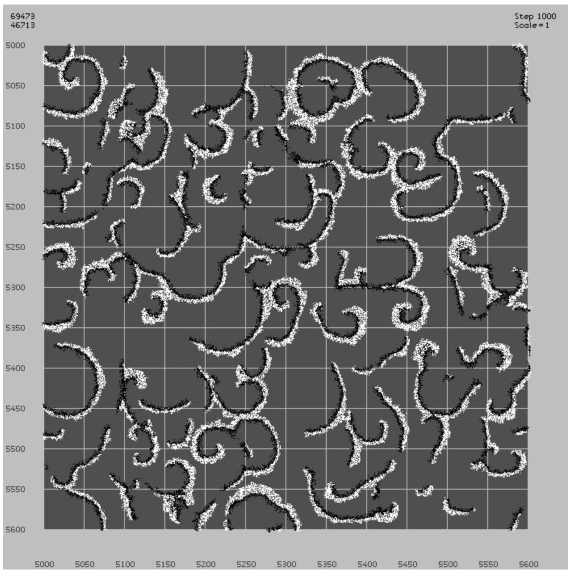

Figure 4: The situation at time step 1000 of a typical run of the predator-prey model. This figure contains over 100,000 *animats* (agents) – predators are black and prey are white. Various macro-patterns, including spiral formations, have emerged.

are fascinated by the emergent flocking behaviours such as that seen in Figure 3.

In each time-step, every individual agent is checked. Each agent has to locate the other agents in its local neighbourhood. Because all agents change position in every time-step, the only way to do this is to check every other agent and this leads to considerable computational overhead.

Typically agents will be stored in some sort of list that has to be traversed to update each one exactly once at each time step.

4 The Animat Model

An agent-based predator-prey model has been developed [31] consisting of "artificial animals" or *animats* [38]. Predators need to consume prey to survive and both species can breed to produce new animats. Animats can "die" from a lack of food or due to "old age" (i.e. if age reaches a set maximum for the species). Prey animats can also be "eaten" by a predator.

In common with most agent-based models, every animat carries a small set of simple rules. The rules are provided as a priority list, i.e. every time step the animat attempts to execute rule 1 first. However each rule has at least one condition that must be satisfied, so if the condition for rule 1 can not be met the animat attempts to execute rule 2 and so on. We have experimented with different rule sets and different priorities of rules (including the evolution of rule sets and priorities over time) [13]. The current rule sets are as follows:

Typical microscopic rules for predator animats:

1. breed if health $> 50\%$ and mate adjacent
2. eat prey if health $< 50\%$ and prey adjacent
3. seek mate if health $> 50\%$
4. seek prey if health $< 50\%$
5. randomly move to adjacent position

Typical microscopic rules for prey animats:

1. breed if health $> 50\%$ and mate adjacent
2. eat grass if health $< 50\%$
3. seek mate if health $> 50\%$
4. move away from adjacent predator
5. randomly move to adjacent position

Note that "mate" refers to any other animat of the same species. Health (along with age) is maintained as part of the state of each animat. Health increases when "food" is consumed and decreases slightly every time step. Grass is placed in various locations around the map – usually evenly spread across most of it. Seek prey/mate causes the animat to look further away (not immediately adjacent) and then move towards a located (and appropriate) animat.

The interaction of the animats as they execute their individual rules has produced interesting emergent features in the form of macro-clusters often containing many hundreds of animats. We have analysed and documented these emergent clusters in [14]. The most fascinating cluster that consistently appears is a spiral as shown in Figure 4.

A list of animats and a full grid structure (two-dimensional array) are maintained. Each animat contains a link to its location in the grid and each grid location contains a link to the animat in that location as depicted in Figure 5.

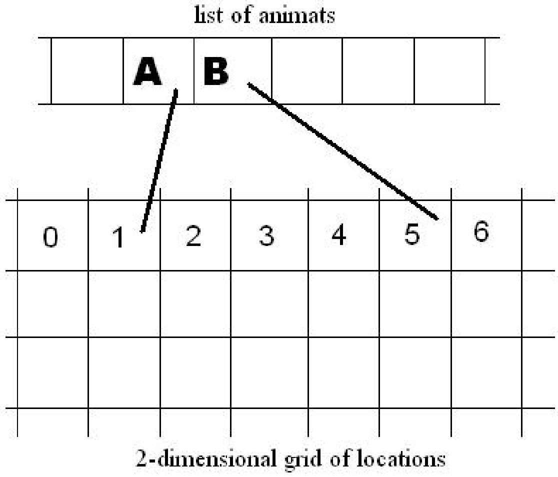

Figure 5: Each animat contains a link to its location in the grid and each grid location contains a link to the animat in that location.

5 Update and Searching

We therefore identify three ways of considering the agent update: a grid-oriented approach ; an individual approach; and a hybrid. The grid approach puts spatial structure as the primary data structure; the individual approach requires storage of agents in a collection such as a list; the hybrid approach uses both with suitable software apparatus to maintain the integrity of both data structures. We describe these below.

5.1 The GRID approach

This is the approach used by classic cellular automata in which the location of each cell in the grid takes precedence over any agents. In fact, agents *per se* do not exist and instead each cell is in one of a number of states. During each time step, every cell in the grid is checked and rules are applied to each cell to determine the state of the cell in the next generation.

Advantages of this approach include: simplicity, regular grid, no links (pointers) required.

Disadvantages include: it is less easy to associate ideas with autonomous individual agents.

5.2 The INDIVIDUAL approach

This is the approach used in Boids in which a list of agents is maintained, each with continuous coordinates. There is no grid because there are no discrete locations. In each time step, every agent has to check all other agents to find which are in its local neighbourhood. This O(n2) approach is enormously costly especially for large numbers of agents.

Various techniques have been used to reduce this overhead including advanced data structures and parallel algorithms. The advanced data structures (such as quad-trees) are difficult to implement and are not particularly effective in a situation where the agents are rapidly changing position [** cite **]. Parallel algorithms can only be used with sophisticated hardware and specialised programming languages.

Advantages of this approach include: independent agents, continuous co-ordinates in 3D space, minimal use of memory (grid is not stored).

Disadvantages include: high computational burden usually limits the number of agents, techniques used to reduce this burden (eg: quad-trees) are complex and are not designed for the rapidly changing locations of the agents.

5.3 The Hybrid COMBINED approach:

This is the approach used in the predator-prey model described in section 4. There is a grid with discrete coordinates and there is also a list of agents. Each agent carries a link to its position in the grid and each grid location carries a link to the agent (or agents) situated in that location as shown in Figure 5.

In each time step, the list of animats is traversed in order to locate the nearest neighbour(s) for each animat by using the following algorithm:

1. follow the link from the animat to the grid to locate the animat's grid location
2. sequence through the lookup table using this grid location as a base
3. use lazy evaluation to set the upper limit

Advantages of the combined approach come from the benefits of grid and individual methods. Identification of neighbouring agents is aided by the grid information but a list agents can easily be traversed separately from the spatial order. For spatial agents the major advantage is the ease of collision tracking both from the perspective of programming and of computational speed and efficiency.

The real disadvantages of the hybrid approach are the need for more sophisticated software and data structures that must be maintained and of course this dual storage approach does also use more memory - which

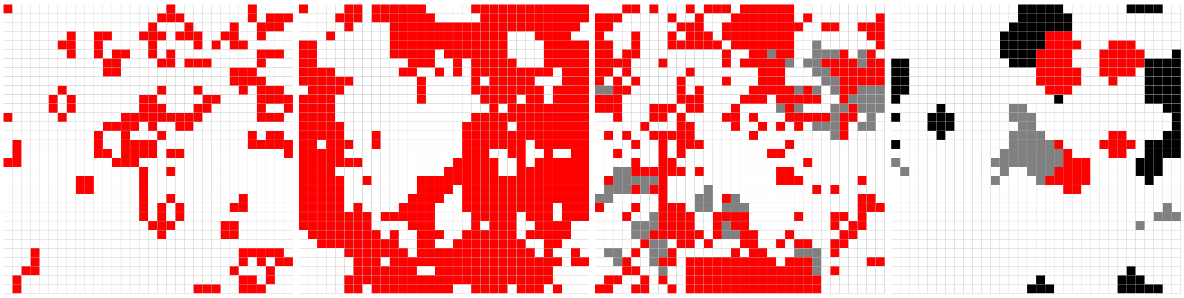

Figure 6: Four models evolved for 100 steps from a random initial mix, on a 32×32 square periodic lattice. From left to right: Game of Life; Ising; Rabbit/Fox Automata [12]; Sznajd Opinion System.

can be a consideration for the very large number of agents typically desired.

6 Discussion of Strategies

We can compare the relative features of agent strategies used in the different models. Figure 6 shows screen snapshots from four of the models discussed: the Game of Life; the Ising model; a Predator/Prey Automata [12] and the Sznajd Opinion agent model. These have varying degrees of self autonomy and make increasing use of hybrid approaches in the list order given.

The grid location provides a huge performance benefit in that the search for neighbours can be carried out by using a lookup table and is thus very rapid. This is shown in Figure 7 where agent "A" traverses its neighbour list in a specific order.

In addition, instead of using a generic "search for neighbours" function, each search is tailored for the rule that is currently being checked. For example, if the agent is a predator with health < 50, the rule checking process would proceed as follows:

1. do not breed as health $< 50\%$
2. search for prey to eat – use only elements 1 to 8 of lookup table as prey must be adjacent (assume this fails for this example)
3. do not seek mate as health $< 50\%$
4. seek prey – continue searching from element 9 of lookup table – stop searching if prey is located and move towards prey
5. randomly move to adjacent position (this rule will only be used if last rule failed)

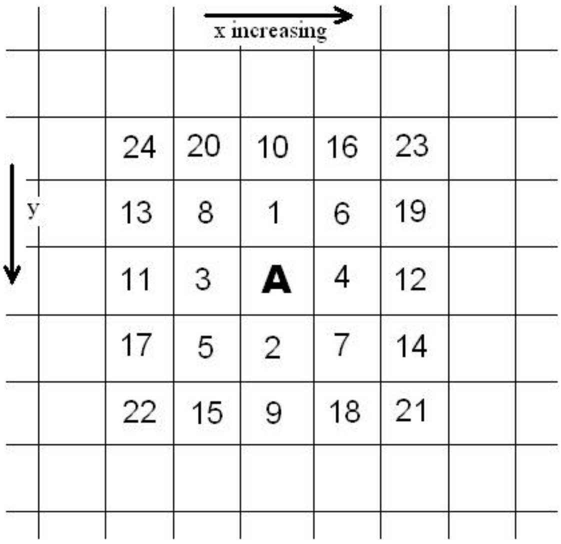

Figure 7: Agent A using the lookup table to locate neighbours. The numbers indicate the order in which the grid will be searched. The first neighbour to be located will be the closest – or at least as close as any other neighbour.

Hence the search during Rule 2 (use only elements 1 to 8) differs from the search in Rule 4 (use elements 9 onwards). This could be regarded as a form of **lazy evaluation**.

A number of sophisticated data optimisation strategies are available to facilitate spatial search though a grid. These include maintaining spatial trees (quad trees in two dimensional systems [30] or oct trees in three dimensions [2,21]); grid-boxing [11]; and other hash table structures [37].

It is our observation that for truly autonomous spatial agent based models it is more convenient and flexible to maintain a hybrid of data structures so that agents can be given the appropriate neighbourhood and other spatial information they require to apply their microscopic rules appropriately.

Traversing a list of N agents is an $\mathcal{O}(N)$ operation. If each agent interacts with each other agent we must apply an $\mathcal{O}(N^2)$ interaction scheme if agents are simply stored in a list, vector or array etc. Hashing optimisation schemes will typically reduce the interactions by some spatial cutoff so that this is reduced to $\mathcal{O}(N \log N)$ but as long as we are prepared to store precomputed neighbour lists and spatial information we obtain the best of both worlds and retain an agent update procedure that is still $\mathcal{O}(N)$, albeit with a larger (bit still linear) prefactor. This is important when we wish to use large numbers of agents $N \rightarrow \approx 10^6$.

7 Conclusion

Through an extensive review of the agent-based model literature we have identified and discussed key differences between the way agent models are formulated conceptually and have reviewed how the split between autonomous individual-based agents and non-autonomous spatial automata leads to important implementation differences in the simulation software required.

In summary, cellular models such as Conway's "Game of life" are homogeneous and spatially densely stored on a discrete lattice or grid based coordinate system. Every cell is always updated deterministically and in the same order. Individual-based agent models such as Reynolds' "Boids" are autonomous and reside in a continuous coordinate space and so are spatially sparse.

Multi-agent based model such as our animat agent system is a hybrid. It uses a coordinate system that is lattice based, but not every lattice cells is occupied and some are multiply-occupied. A hybrid approach that tracks a list of animat agents where each links or points to a spatial location is possible.

It is therefore feasible to obtain significant computational optimisations by storing spatial cell neighbourhood information once as part of the grid structure while allowing autonomous agent update s to access that information at the appropriate update point.

We are investigating the use of a spatial hash structure so that neighbours lists can be maintained alongside continuous spatial agent coordinates that "hash into" the approximate grid structure. The dual approach appears to have great potential for imporved computational performance and hence system size scalability. Spatial multi-agent system will continue to find uses in a range of applications that require large system sizes to be simulated in order to expose multi-scale emergent effects. We therefore believe that these techniques will be of even higher importance for future agent-based simulations of realistically sized systems.

References

[1] Adami, C.: On modeling life. In: Brooks, R., Maes, P. (eds.) Proc. Artificial Life IV. pp. 269–274. MIT Press (1994)

[2] Bedorf, J., Gaburov, E., Zwart, S.P.: A sparse octree gravitational n-body code that runs entirely on the gpu processor. J. Comp. Phys. Online December, 1–34 (2011)

[3] Cares, J.R.: The use of agent-based models in military concept development. In: Proc. 2002 Winter Simulation Conference. pp. 935–939. San Diego, California, USA (8-11 December 2002)

[4] Cioppa, T.M., Lucas, T.W., Sanchez, S.: Military applications of agent-based simulations. In: Proc. 2004 Winter Simulation Conference. pp. 171–180. Washington DC, USA (5-8 December 2004)

[5] Cornforth, D., Green, D.S.: Intelligent Complex Adaptive Systems, chap. Modularity and Complex Adaptive Systems, pp. 75–104. IGI Global (2008)

[6] da Silva, B.C., Junges, R., de Oliveira, D., Bazzan, A.L.C.: Itsumo: an intelligent transportation system for urban mobility. In: AAMAS '06: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems. pp. 1471–1472. ACM, New York, NY, USA (2006)

[7] Gardner, M.: Mathematical Games: The fantastic combinations of John Conway's new solitaire game "Life". Scientific American 223, 120–123 (October 1970)

[8] Haddawy, P., Dhananaiyapergse, K., Kaewpitakkun, Y., Bui, T.: Data-driven agent-based simulation of commercial barter trade. In: Proc. IEEE/WIC/ACM Int. Conf. on Intelligent Agent Technology (IAT'04) (2004), iSBN 0-7695-2101-0/04

[9] Hawick, K.A.: An agent model formulation of the ising model. Tech. rep., Information and Mathematical Sciences, Massey University, Albany, North Shore 102-904, Auckland, New Zealand (November 2003)

[10] Hawick, K.A., James, H.A., Scogings, C.J.: Roles of rule-priority evolution in animat models. In: Proc. Second Australian Conference on Artificial Life (ACAL 2005). pp. 99–116. Sydney, Australia (December 2005)

[11] Hawick, K.A., James, H.A., Scogings, C.J.: Gridboxing for spatial simulation performance optimisation. In: T.Znati (ed.) Proc. 39th Annual Simulation Symposium. Huntsville, Alabama, USA (April 2006), the Society for Modeling and Simulation In-

ternational, Pub. IEEE Computer Society

[12] Hawick, K.A., Scogings, C.J.: A minimal spatial cellular automata for hierarchical predator-prey simulation of food chains. In: International Conference on Scientific Computing (CSC'10). pp. 75–80. WorldComp, Las Vegas, USA (July 2010), www.massey.ac.nz/~kahawick/cstn/040

[13] Hawick, K.A., Scogings, C.J.: Spatial pattern growth and emergent animat segregation. Web Intelligence and Agent Systems 8(2), 165–179 (2010)

[14] Hawick, K.A., Scogings, C.J., James, H.A.: Defensive spiral emergence in a predator-prey model. Complexity International (msid37), 1–10 (October 2008), http://www.complexity.org.au/ci/vol12/msid37, ISSN 1320-0682

[15] Hawick, K.: Multi-party and spatial influence effects on opinion formation models. In: Proc. IASTED International Conference on Modelling and Simulation (MS 2010). No. CSTN-032, Calgary, Canada (June 2010), paper 696-035

[16] Hawick, K.: Cycles, diversity and competition in rock-paper-scissors-lizard-spock spatial game simulations. In: Proc. International Conference on Artificial Intelligence (ICAI'11). Las Vegas, USA (July 2011)

[17] Hawick, K., Scogings, C.: Cycles, transients, and complexity in the game of death spatial automaton. In: Proc. International Conference on Scientific Computing (CSC'11). No. CSC4040, Las Vegas, USA (July 2011)

[18] Holland, J.H.: Echoing emergence: Objectives, rough definitions, and speculations for echo-class models. In: Cowan, G.A., Pines, D., Meltzer, D. (eds.) Complexity: Metaphors, Models and Reality, pp. 309–342. Addison-Wesley, Reading, MA (1994)

[19] Husselmann, A., Hawick, K.: Simulating species interactions and complex emergence in multiple flocks of boids with gpus. In: Proc. IASTED International Conference on Parallel and Distributed Computing and Systems (PDCS 2011). pp. 100–107. Dallas, USA (14-16 Dec 2011)

[20] Ising, E.: Beitrag zur Theorie des Ferromagnetismus. Zeitschrift fuer Physik 31, 253?258 (1925)

[21] Jackins, C.L., Tanimoto, S.L.: Oct-trees and their use in representing three-dimensional objects. Computer Graphics and Image Processing 14, 249–270 (1980)

[22] King, A.J., Streltchenko, O., Yesha, Y.: Using multiagent simulation to understand trading dynamics of a derivatives market. Annals of Maths and AI 44(3), 233–253 (July 2005)

[23] Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H.., Teller, E.: Equation of state calculations by fast computing machines. J. Chem. Phys. 6(21), 1087–1092 (1953)

[24] Owen, M., Lee, L., Sewell, G., Steward, S., Thomas, D.: Multi-agent trading environment. BT Technology Journal 17(3), 33–43 (July 1999)

[25] Radecký, M., Gajdoš, P.: Intelligent agents for traffic simulation. In: SpringSim '08: Proceedings of the 2008 Spring simulation multiconference. pp. 109–115. The Society for Computer Simulation, International, San Diego, CA, USA (2008)

[26] Reynolds, C.: Flocks, herds and schools: A distributed behavioral model. In: SIGRAPH '87: Proc. 14th Annual Conf. on Computer Graphics and Interactive Techniques. pp. 25–34. ACM (1987), iSBN 0-89791-227-6

[27] Reynolds, C.: Boids background and update (July 2011), www.red3d.com/cwr/boids/

[28] Reynolds, C.W.: Steering behaviors for autonomous characters. In: Proc. Game Developers Conference, San Jose, CA, USA. pp. 763–782 (1999)

[29] Ronkko, M.: An artificial ecosystem: Emergent dynamics and lifelike properties. J. ALife 13(2), 159–187 (2007)

[30] Samet, H.: Data structures for quadtree approximation and compression. Communications of the ACM 28(9), 973–993 (1985)

[31] Scogings, C.J., Hawick, K.A., James, H.A.: Tools and techniques for optimisation of microscopic artificial life simulation models. In: Nyongesa, H. (ed.) Proceedings of the Sixth IASTED International Conference on Modelling, Simulation, and Optimization. pp. 90–95. Gabarone, Botswana (September 2006)

[32] Standish, R.K.: Intelligent Complex Adaptive Systems, chap. Concept and Definition of Complexity, pp. 105–124. IGI Global (2008)

[33] Sznajd-Weron, K.: Sznajd model and its applications. Acta Physica Polonica B 36(8), 2537–2547 (2005)

[34] Sznajd-Weron, K., Krupa, S.: Inflow versus outflow zero-temperature dynamics in one dimensional. Phys. Rev E 74(74), 031109–1–8 (2006)

[35] Trautteur, G., Virgilio, R.: An agent-based computational model for the battle of trafalgar: A comparison between analytical and simulative methods of research. In: Proc. Twelfth Int. Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE'03). pp. 377–382. Linz, Austria (9-11 June 2003), iSBN:0-7695-1963-6

[36] Tyrrell, T., Mayhew, J.E.W.: Computer simulation of an animal environment. In: Meyer, J.A., Wilson, S.W. (eds.) From Animals to Animats, Proceedings of the First International Conference on Simulation of Adaptive Behavior. pp. 263–272 (1991)

[37] Warren, M.S., Salmon, J.K.: A parallel hashed oct-tree n-body algorithm. In: Supercomputing. pp. 12–21 (1993), citeseer.ist.psu.edu/warren93parallel.html

[38] Wilson, S.W.: The animat path to AI. In: Meyer, J.A., Wilson, S. (eds.) From Animals to Animats 1: Proceedings of The First International Conference on Simulation of Adaptive Behavior. pp. 15–21. Cambridge, MA: The MIT Press/Bradford Books (1991)

[39] Wolfram, S.: Statistical Mechanics of Cellular Automata. Rev.Mod.Phys 55(3), 601–644 (1983)

[40] Wolfram, S.: Theory and Applications of Cellular Automata. World Scientific (1986)